Exploring the Effects of View Transforms on Self-Supervised Video Representation Learning Techniques

Ilian Herzi
David Chan, Ed.
John F. Canny, Ed.

Electrical Engineering and Computer Sciences
University of California, Berkeley

May 18, 2021

Acknowledgement

Exploring the Effects of View Transforms on Self-Supervised Video Representation Learning Techniques

by

Ilian H. Herzi

A **Master of Science, Plan II** submitted in partial satisfaction of the

requirements for the degree of

Master's of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor John F. Canny: Advisor, Research Advisor
Professor Trevor Darrell

Spring 2021

Exploring the Effects of Spatiotemporal View Transforms on Self-Supervised Video Representation Learning Techniques
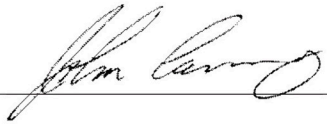
by Ilian H. Herzi

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II.**

Approval for the Report:

Research Advisor _____   Date   5/13/2021

Professor John F. Canny

Second   Reader _____   Date   5/17/21

Professor Trevor Darrell

University of California, Berkeley

Exploring the Effects of View Transforms on Self-Supervised Video Representation Learning Techniques

Abstract

Exploring the Effects of View Transforms on Self-Supervised Video Representation Learning Techniques

by

Ilian H. Herzi

Master's of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor John F. Canny: Advisor

Self-supervised video representation learning algorithms, such as pretext task learning, contrastive learning, and multimodal learning, have made significant progress in extracting features that generalize well to downstream video benchmarks. All of these learning algorithms rely on the underlying view transforms and research on how view transformations impact the performance of these learning algorithms has not been thoroughly explored. In this work, we investigate the effect of many different spatial, temporal, and visual view transforms on pretext task learning and contrastive learning. We provide a detailed analysis of the performance of these methods on video action recognition, and investigate how different methods compare by combining the learned features of several models pretrained using different learning algorithms and/or view transforms. In our setup, certain combinations of pretraining algorithms and view transforms perform better than supervised training alone on the UCF-101 and HMDB action recognition datasets but underperform some of the current state-of-the-art methods.

I dedicate this to my brother, my sister, my mother, and my father. To my brother who stands tall and has faced his adversities head on, my sister who continues fighting in the face of darkness, my mother who grows like the plants she tends, and my father who has stayed strong and loving as he dealt with loss. Lazare and Leila, you make me proud to be your brother. Mom and Dad, you make me proud to be your son.

I also dedicate this to my friends, dysf.e., the crusin crusaders, french central station, and the ones who help me grow.

And to my advisor John F. Canny and my mentor David M. Chan.

With all of you I stand taller.

Contents

List of Figures

List of Tables

Acknowledgments

I'd like to acknowledge my advisor Professor John F. Canny for his help with this work, my mentor David M. Chan for his insight, conversations, and help with this project without whom this wouldn't have been possible. I'd also like to thank the members of AAI in Canny Lab, notably Aatif Jiwani, Sumanth Gurram, Oliver Bryniarski, Andy Fang, and Adrian Liu for their feedback and helpful conversations.

Chapter 1

Introduction

Videos are ubiquitous in daily life. Sharing content through videos has quickly become the greatest source of internet traffic [51]. Finding solutions that can parse videos and output the desired insight has been one of the biggest areas of research over the past few years. Video domain problems pose a significant scale difference compared to their image domain counterparts due to the increased number of images per video and the addition of audio data and sometimes language data. The difference in size can make labeling and interpreting videos by hand expensive. One way to ameliorate this cost is to use unsupervised learning techniques on large unlabeled datasets to learn features that generalize well to video downstream tasks.

Purely self-supervised representation learning methods on some benchmarks are competitive and sometimes better than purely supervised approaches and approaches that pretrain on a larger dataset and then finetune. These state-of-the-art (SOTA) approaches predominately use pretext task learning, contrastive learning, or multimodal learning which we elaborate on in Section: 2 and Section: 5.

Pretext training algorithms and contrastive learning algorithms rely heavily on the transforms used to create views, which for this paper we call *view transforms*. Intuitively, each is used to define a pseudo-equivalence relation that depends on the learning algorithm: pretext tasks relate transform realizations and attempts to be instance invariant while contrastive learning relates instances and attempts to be view transform invariant.

Recent work has analyzed how view transforms impact these learning algorithms. Notably, Wu *et al.* [66] analyze representation learning using the mutual information between views but their experiments analyzed only 5 view transforms. Some work has also explored combining these learning algorithms into a pretext + contrastive learning algorithm [6, 54]. However, experiments on how each view transform changes the performance of these learning algorithms and how they interact in these frameworks is largely unexplored.

Learning what kind of view transform works best for each of these modern approaches can help guide practitioners on the best view to use when using one of these algorithms. Using the empirical results of this thesis, we hope to add insight to future work in representation learning.

In this work, we specifically investigate different spatial, temporal, spatiotemporal, visual view transforms, and their combinations in the video action recognition task using pretext learning, contrastive learning, and briefly pretext + contrastive learning pretraining. We find that Rotation by 90° tends to outperform all pretext task methods while our new view transform, FFT, inspired by human perception [57], outperforms all contrastive learning methods on video action recognition benchmarks. We also investigate combining independently learned features in the latent space and generally do better than a purely supervised method in our setup.

We also briefly investigate the effect of modifying the underlying backbone in the learning algorithm and find that performance depends heavily on the pretraining algorithm and downstream learning task.

Chapter 2

Related Works

Current work in unsupervised video representation learning algorithms can be broken down into roughly three areas of research: methods that rely on pretext tasks, contrastive learning, and multimodal learning (which is usually a subset of the other two learning algorithms). Each of these techniques depends on a view transform that extracts a specific representation from the video (note in the case of multimodal this is naturally restricted to modality views as defined in Section: 3.1).

For pretext task learning, most methods essentially design novel view transforms to extract meaningful labels to train on. Some pretext tasks in the literature exploit spatial views and inspire some of the view transforms described in Section: 3.2. One such method generates jigsaw puzzles, a.k.a. Spatial Jigsaw views, that creates spatial patches in each frame and shuffles them [1, 40, 64]. Doersch *et al.* use this Spatial Jigsaw view in a pretext task that predicts the spatial order of patches [17]. Work by Jing *et al* explore rotations as a view transform, explicitly predicting rotations [25]. Some pretext tasks are designed specifically for videos and exploit their temporal views and inspire some of the view transforms described in Section: 3.3. Work by Lee *et al.* shuffle the frames and try to predict the order [31], which is a pretext task that uses a Temporal Jigsaw view with a temporal chunk size of 1 and is analyzed in this work. Other temporal pretext tasks predict relative video speeds, or Pace views, and is similar to work from Wang *et al.*, Cho *et al.*, Yao *et al.* [59, 14, 69]. Wei *et al.* focus on the nature of time and explicitly try to model features in the forward temporal direction [65]. We hypothesize that learning a single direction of time can help with most downstream tasks so long as these temporal views capture motion information about the videos, which we explore in the InvertOrder transform Section: 3.3. Recent pretext methods in video representation learning have proposed using both the spatial and temporal views together exemplified by work from Kim *et al.* [28] which tries to order Spatiotemporal puzzles and Wang *et al.* [60] which tries to predict the motion diversity and color diversity of a video.

Other works instead randomly select from a set of view transforms and try to predict which transform was used. Jenni *et al.* create a set of Pace view transforms and attempt to predict what Pace transformation was used; this is actually quite similar to the afore-

mentioned Pace view prediction pretext task [59, 14, 69]. Additionally, work from Luo *et al.* [35], attempt to predict spatial and temporal view transforms as a pretext task. Note though, that all of these predicting-the-transformer based pretext tasks use carefully selected view transforms and which combinations of transforms work best for these kinds of tasks is still unclear. This uncertainty in part helps motivate our exploration of view transforms. However, since this method is a high-order view transform and requires considering combinations of views we leave this investigation to future work and instead focus on the independent influence of single views on other pretraining algorithms.

In contrastive learning, some works explore modifications to the contrastive loss such as NT-Xent in SimCLR [12]. Because exploration of contrastive losses are not the focus of this paper we select NT-Xent for all our contrastive experiments. What makes a 'good' view for contrastive learning has also been examined in the literature. Tian *et al.* [55] demonstrate that the best views for a given contrastive task should contain information that's relevant to the downstream task while minimizing noise between views. Other work has empirically investigated the effectiveness of temporal view transforms that sample clips from both local and global temporal regions of the video [16, 33, 61]. We expand on their findings by analyzing more view transforms and adding to their contributions. The pace related views used in pretext training can also be used contrastive learning, where a pace consistency is contrastively enforced between video clips [68]. Similar to the innovations in pretext task learning, other spatiotemporal views have been investigated in the contrastive learning framework. RSPNet [11] is trained contrastively using an alignment between pace and spatial features on sampled clips. Other methods like CVRL [47] sample clips from a video that vary in temporal distance and then transform those views spatially essentially composing the two view transforms into a spatiotemporal transform. To reiterate, while we consider a few inherently spatiotemporal view transforms like Spatiotemporal Jigsaw 3.4, we restrict our experiments to ones that are single view transforms and not compositions of transforms.

While we do not explicitly test the following approaches we briefly touch upon other SOTA pretraining methods that have performed well on downstream video tasks. Since contrastive learning acts on instances, some work has investigated operating on clusters, coined prototypical contrastive learning, which learns clusters and optimizes the contrastive loss and clusters in an E-M like manner [32].

A significant progress in video related benchmarks have used multimodal learning. Most multimodal approaches are extensions of pretext or contrastive learning techniques that use paired views from data domains other than video such as audio and text. Some work contrastively learns from video-audio, and video-language data pairs [4, 39, 50, 3, 36, 45, 38, 37]. Recent multimodal approaches have achieved SOTA performance on video action recognition datasets, notably Akbari *et al.* [2] which uses a video-audio text transformer that incorporates video, audio, and text simultaneously, and CLIP [48] which jointly trains an image feature extractor with a text encoder, altogether ignoring the temporal element of videos. Patrick *et al.* have proposed general approaches to multimodal learning that consider all view transforms as hierarchical multimodal, spatiotemporal, generalized transforms in the

context of contrastive learning [44].

There is also work that uses an optical flow or motion view transform to extract some sort of motion information and some of these approaches have also shown SOTA on video action recognition [62]. We choose not to use optical flow because if motion information is necessary for downstream tasks then a learning algorithm should extract this information implicitly.

Other methods borrow techniques from masked language modeling (MLM) and use vector sequences as representations of videos. These representations are then passed into language architectures like BERT and trained contrastively [26]. Others use paired language and video data in this MLM-like framework to encode representations, like in VideoBERT [53].

More recent work has explored combining pretext and contrastive learning methods, notably in TaCo [54] and Bai *et al.* [6], which balance these two objectives and achieve results that outperform either of the individual training methods used in isolation.

For this work we primarily focus on view transforms and their influence on pretext training, contrastive training, and pretext + contrastive training. We do not wish to ignore progress in multimodal learning, however as the view transforms for multimodal learning are well-defined we instead fixate on algorithms that utilize different views.

Chapter 3

Methods

In this section, we give an overview of the different view transforms explored and later in Section: 4 we motivate our general pretraining setup and our experimental setup. Our framework consists of 1.) selecting a view transform 2.) selecting a pretraining method, and 3.) evaluating the learned features by finetuning them to our target downstream tasks, which we also evaluate against prior work. This section describes the views while in Section: 4.8 we describe the details of the learning algorithms, datasets, architectures, and implementation for each of these steps. Results and a comparison of our work is listed in Chapter: 5.

3.1   Views

As mentioned earlier, the success of self-supervised representation learning algorithms depends heavily on the view transform. In this chapter, we discuss all three types of views that can be applied to the video data domain. In Figures: 3.1, 3.2, 3.3, and 3.4 we visualize some of the transforms used in our experiment. A "*" indicates transforms that were considered in this work but not adequately tested and should be explored in the future. Note that this list is not exhaustive. For clarity, we define some key terms that are used throughout this thesis. A video $V \in \mathbb{R}^{(S,H,W,C)}$. A *view transform*, $\mathcal{T}$, is a set of N possible transformations, T_i s.t. $\mathcal{T} = \{T_1, ..., T_N\}$, where each transform is a map $T_i : \mathbb{R}^{(S,H,W,C)} \rightarrow \mathbb{R}^{(S',H',W',C')}$. We refer to T_i as a *realization* of a view transform and an element $V' \in \mathbb{R}^{(S',H',W',C')}$ as the generated *view* under $\mathcal{T}$.

3.2   Spatial Views

Spatial views are generated by transformations that operate on the spatial dimensions of the video (per-frame) and are designed to capture single-moment spatial relationships between the data. For all of the spatial views, each transformation is applied consistently across all frames. (i.e. a rotation by 90° would imply that each frame in the video is rotated by 90°.)

Random Crop: The Random Crop transform follows the approach in SimCLR [12], where a crop is randomly chosen from some set of possible crops and then applied consistently to all frames in a video. The sizes of the possible cutouts are specified apriori.

Random Cutout: In the Random Cutout transform, a random patch is 'cutout' by masking that patch to zero. Each cutout is randomly chosen from some set of possible cutouts. The sizes of the possible cutouts are specified apriori.

2D-Fast Fourier Transform: Inspired by the De Valois model of visual perception [57], the 2D-Fast Fourier Transform converts a spatial frame to a frequency frame using the discrete Fast Fourier transform (DFFT) (with OpenCV and NumPy [8, 22]). Different parts of the spatial frequencies are masked according to a random circular mask (selecting explicitly high or low frequencies) or torus mask (selecting or excluding a band of frequencies). The size of each mask is restricted to be less than the size of the frame. The masked frequency frames are then converted back to the space domain. Each mask is applied consistently across frames. This transform is a spatial filter. See Fig 3.1.

Flip: The Flip transform randomly flips each video frame either horizontally, vertically, both, or neither (identity transform). See Fig 3.1.

Rotation: Rotation by some random angle R takes each frame within a video and rotates the frame around its center by the same random R sampled from $[0, 360]$ degrees. We discretize the angle of rotation into $\frac{360}{R}$ bins and transform only according to the bins. Any missing pixels are filled with black pixels. See Fig 3.1.

Shear: The Shear transform randomly shears each frame within a video along the x and/or y-axis by some shear angle amount in $[-0.5, 0.5]$ generating a random affine transformation matrix. We discretize the shear value into $\frac{1}{S}$ bins to restrict the total number of shears. For example, the "Shear$_{x,y}$ .25" transform consists of 16 bins, since we take the Cartesian product of all possible shears along the x and y directions. See Fig 3.1

Spatial Jigsaw (Permutation or Relative): The Spatial Jigsaw transform, inspired by work from Unaiza *et al.* and Noroozi *et al.* [1, 23], divides each video frame into a grid of $\sqrt{P} \times \sqrt{P}$ spatial chunks. These spatial chunks are randomly permuted and rasterized together to maintain the original frame shape and global context. The number of possible transformations is either the total number of permutations of the chunks $P!$ or the total number of chunks per chunk where each chunk position is independently predicted with a label corresponding to one of P labels in the grid. See Fig: 3.1.

Translate: In the Translate transform, each video frame is translated along the x and/or y-axis by a random translation amount sampled from $[-150, 150]$. We discretize the translation value into $\frac{300}{T}$ bins. All missing pixels following translation are filled with black pixels. See Fig: 3.1.

Translate+WrapAround:* Identical to the Translate transform except black artifacts are filled with the image pieces cutout by the translation.

Translate+Zoom: In the Translation+Zoom transform, each video frame is randomly translated in one of 9 directions relative to the center (N, NW, W, SW, S, SE, E, NE, center (no translation)). Because translation leaves black pixels as artifacts, once the frame is

translated, the image is enlarged until the scaled frame fills the border of the original frame shape whereupon it is cropped back to the original frame shape.

Zoom: In the Zoom transform, each video frame is randomly zoomed by a scale that's randomly selected from a finite set of Z scales. See Fig: 3.1.

3.3 Temporal Views

Temporal views are generated by transforms on the temporal axis of videos. Inspired by work in [59, 14, 69, 47] we explore several temporal view transforms.

Invert Order: In the Invert Order transform, the frames of a video are randomly reversed. See Fig: 3.2.

Pace: In the Pace transform, the relative frame rates are either accelerated or slowed down with respect to the original clip's frame rate by sampling at different *paces*, i.e. rates, following the description by Wang *et al.* [59], but is also similar to the work Cho *et al.* and Yao *et al.* [14, 69]. A pace equal to 1 is normal motion and a pace greater than 1 is fast motion. We sample every pth frame where and a pace less than 1 duplicates $\frac{1}{p}$ frames. (i.e. for Pace $\{.5, 1, 2, 3, 4\}$ we randomly select a pace, p and sample every p-th frame if p is > 1 or duplicate if p is < 1). See Fig: 3.2.

Temporal Jigsaw: In the Temporal Jigsaw transform the original video is divided into T temporal chunks with an equal number of frames in each chunk. The chunks are then permuted along the temporal axis. With T=1 this becomes equivalent to sequence sorting seen in [31]. See Fig: 3.2.

Start Prediction*: The Start Prediction transform rotates the indices of a video and attempts to predict where the start index is. It can be seen as a special case of the Temporal Jigsaw transform.

Video Clip Order Prediction (VCOP)*: Video Clip Order Prediction follows the methodology in work from Xu *et al.* [67] where given a video, clips at different temporal locations are sampled and then reordered. The original goal of this task is to then predict the correct temporal order of these clips however this view can be defined generally.

3.4 Spatiotemporal Views

Spatiotemporal transforms generate views by considering both the spatial and temporal axis of a video simultaneously.

Spatiotemporal Jigsaw: Inspired by STPuzzle from Kim *et al* [28], Spatiotemporal Jigsaw combines Spatial Jigsaw and Temporal Jigsaw, permuting 3D patches within a video. See Fig: 3.3.

Temporal Clip Sampling*: Temporal Clip Sampling follows the methodology of CVLR from Qian *et. al* in [47] where two different clips from the same video are sampled at different temporal locations, usually before applying some other view transform.

3.5   Visual Views

Visual views are generated by transforms that are applied at a pixel-level. These transforms do not explicitly change the relative spatial or temporal relationships in the frames. Note, that if the transforms are uniform at a pixel level (such as a brightness transform), a model using batch normalization will compensate automatically.

Auto Contrast: In the Auto Contrast transform, frames either have their contrasts normalized to map their min pixel value to 0 and their max pixel value to 255 or not modified at all.

Color Jitter: In the Color Jitter transform, frames are randomly composed with four transforms: brightness, contrast, saturation, and hue. Brightness is the relative lightness of a color, contrast is the magnitude of the difference in luminance, saturation is the intensity of a hue, and a hue is a spot on the color wheel. For each of these four transforms the parameters are discretized by C in the range $[.1, 1.1]^4$. See Fig: 3.4.

Edge Filters: For Edge Filter transforms, a set of filter transformations are selected beforehand and all frames within a video are transformed by a filter randomly selected from this set. (i.e. a video can be randomly transformed by the identity, a Canny filter, the Sobel filter, the Scharr filter, the Roberts filter, the difference of Gaussians, the Farid filter, or the Laplace transform.) See Fig: 3.4.

Invert*: The Invert view transform reverses RGB values into their complementary colors on the color wheel. See Fig: 3.4.

Equalize*: The Equalize view transform applies a nonlinear map to create a uniform distribution of greyscale values in the image. See Fig: 3.4.

Solarize*: The Solarize view transform inverts all pixel values above some threshold, T. See Fig: 3.4.

Posterize*: The Posterize view transform quantizes the the integer bit representation of the RGB values to a random bit size from $\{2, 4, 8\}$. See Fig: 3.4.

Style GAN*: Inspired by considering view generation as an video-to-video problem, The Style GAN transform randomly selects an image-to-image translation model that has been trained on a style from S possible style domains. All frames within a video are transformed consistently by the translation model.

Multimodal Views

Multimodal views are transformations on paired data from multiple modalities or data that was collected from the same source e.g. a video and an audio track. Any transform that extracts something from paired data is considered a multimodal view.

Video Embeddings*: In the Video Embeddings transform, a video paired with some audio and language data is transformed by a feature extractor that compresses the video into some feature vector.

Audio Embeddings*: In the Audio Embeddings transform, sounds associated with some video and language data are extracted and transformed into spectral frequencies using

the Fourier transform. The spectrograms are then transformed into some audio feature vector.

Language Embeddings*: In the Language Embeddings transform, labels or descriptions paired with some video and audio data are extracted and transformed by a feature extractor into some language embedding.

Optical Flow*: The Optical Flow transform uses dense optical flow to capture motion features between frames of the video.

Other views

Other views that have shown success in self-supervised pretraining on videos but deviate from the spatial, temporal, visual, and multimodal view categorization are as follows:

Counting*: In the Counting transform, counts of different visual primitives in both the frames and patches of the frames are extracted. Inspired by Noroozi *et al.* [41], the original task enforced equality between the count of frame visual primitives and the sum of detected visual primitives in patches of the frame.

Object Recognition Labeling*: Using pretrained 2D object detection models to label the instances in the images. A variation of enforcing visual primitives.

Spatiotemporal Statistics*: In the Spatiotemporal statistics transform, motion and color diversity statistics are extracted from a video. Inspired by work from Wang *et al.* [60] and Wang *et al.* [62].

Transform Recognition*: Inspired by work from Jenni *et al.* and Luo *et al.* [24, 35], a set of view transforms, $\mathcal{T}_0...\mathcal{T}_M$, is selected beforehand. A view is generated by first randomly selecting a view transform from $\mathcal{T}_0...\mathcal{T}_M$ and then applying it on the video. This view transform is a higher-order function that takes in arbitrary counts of other view transforms.

Figure 3.1: Spatial Transformations
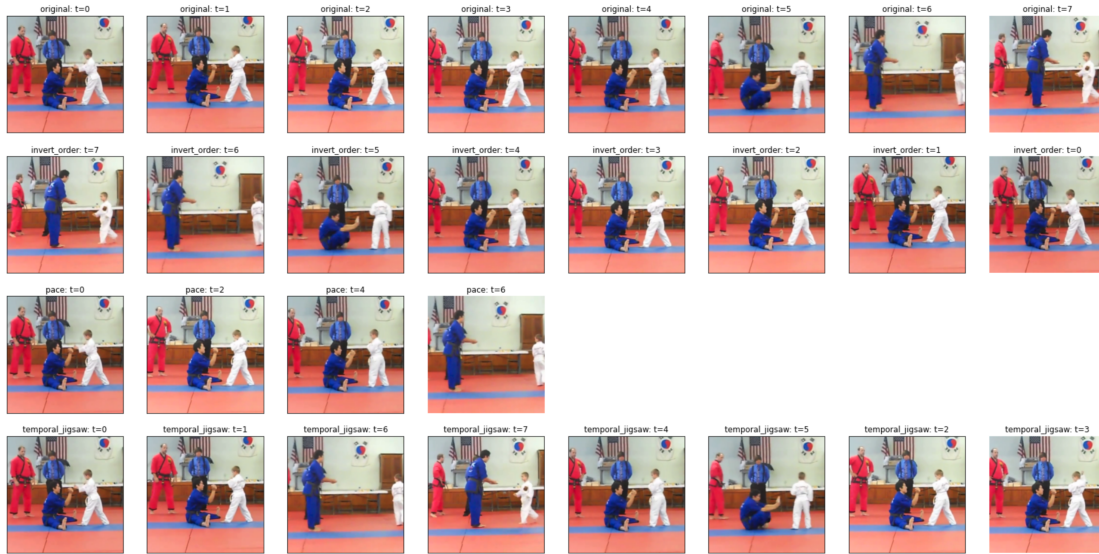
Figure 3.2: Temporal transformations

Figure 3.3: Spatiotemporal transformations
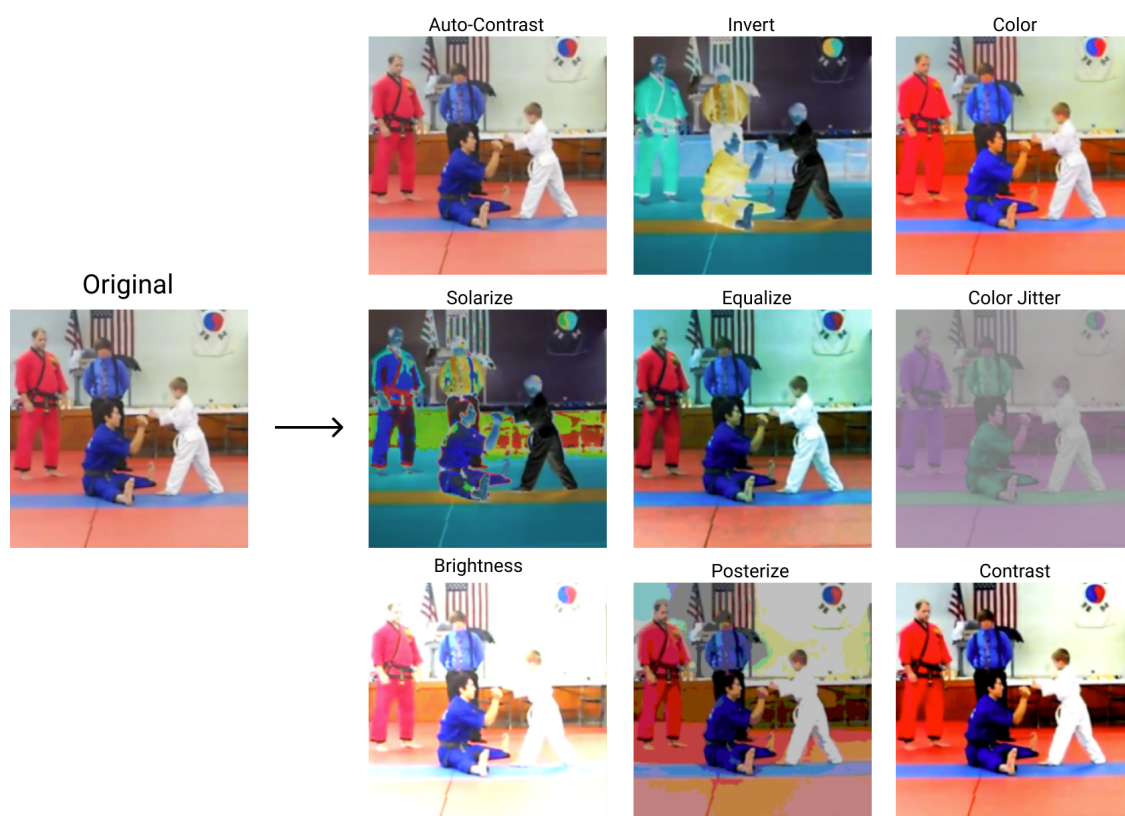
Figure 3.4: Visual transformations

Chapter 4

Experimental Design

In this section, we continue from the above (Section: 3) and describe the different pretraining algorithms used, the datasets used, the architecture used, and the setup for each experiment.

4.1 Pretext Training

The goal of pretext task-based self-supervised learning is to define a task for which labels can be generated from view transforms (such as the ones listed in Section 3.1) usually on an unlabeled dataset. The learning signal that trains the model comes from differences in the selected view transform realizations and ideally this signal should contain enough information to teach the model to extract features that can be used effectively in downstream tasks. (See Fig 4.1.)

Formally, for a video V, and a view transform $\mathcal{T} = T_0, ..., T_N$, we generate several transformed views of the video $T_0(V), ..., T_N(V)$. Then we train a model, $f(T_i(V), \Theta)$, to predict the target label i from the transformed view of the V, yielding some loss $L_{pretext}$ that's optimized. After the model is trained to perform this task, researchers either use the mid-level features of this model, f, (e.g. the last layer before prediction)[13] as a representation of the video or they use the trained parameters, Θ, to initialize a new fully trainable model for the downstream task [25]. We opt for the former method and discuss this decision in Section: 4.8. To reiterate, pretext tasks can be thought of as a pseudo equivalence relation that enforces equivalence between different view realizations across video instances. (See Fig. 4.1.)

4.2 Contrastive Training

Formally, for a video V, the identity function Id, and a view transform $\mathcal{T} = T_0, ..., T_N$ we train a model f with parameters Θ that attempts to maximize the mutual information between the $Id(V)$ and $T_i(V)$ as shown in Equation: 4.1. Some work in contrastive learning

has investigated optimizing lower bounds on the mutual information, such as the max-margin contrastive loss, the triplet loss, NT-Xent loss [12], and InfoNCE loss. Each of these objectives can be used for the contrastive loss, $L_{contrastive}$, to optimize the model. We opt for the NT-Xent loss and discuss why in Section: 4.8.

$$\max_{\Theta} I(f(Id(V), \Theta), f(T_i(V), \Theta)) \ \forall V \tag{4.1}$$

Once an objective is picked and a model trained, similar to the pretext scenario researchers either use mid-level representations of this model [13] or use it to initialize a fully trainable model for the downstream task. Intuitively, contrastive tasks can be thought of as pushing different view transform realizations for a single instance closer together in a latent space and repelling different instances via negative sampling as discussed in work from Wang *et al.* [63].

Note, the importance of the mutual information between the generated views should not be ignored. Lots of work analyzes the ideal amount of shared mutual information during contrastive learning for optimal downstream performance; indeed, Tian *et al.* show that ideal views share high downstream task-related information and low noise and Wu *et al.* give theoretical conditions on good views [55, 66]. However, work by Tschannen *et al.* [58] have shown that accuracies in downstream performance do not align completely with mutual information scores as accuracies in downstream performance can be improved while fixing the mutual information, and the mutual information can be improved while fixing the accuracy. These conflicting observations emphasize the need to empirically analyze multiple views in the context of these different pretraining algorithms. (See Fig 4.2 for an overview of contrastive learning.)

4.3 Pretext + Contrastive Training

In Pretext + Contrastive Training we combine both the pretext and contrastive learning objectives, simultaneously optimizing over both algorithms by combining their loss as shown in Equation: 4.2. α is a hyperparameter that balances the two losses following the methods in TaCo [6] where performance improves as the losses of the two learning algorithms are balanced. Combining these two tasks does not intuitively guarantee any particular gains since pretext training seems to aggregate instances while contrastive learning seems to separate instances. However, [6, 54] empirically show improved accuracies on downstream tasks when these two objectives are combined. In our work, we very briefly explore this pretraining regime with our best transforms for both learning algorithms. (See Fig 4.3)

$$L_{pcl} = L_{contrastive} + \alpha L_{pretext} \tag{4.2}$$

4.4   Combination of pretraining tasks

There are several ways to combine pretraining tasks for learning strong representations, and in this experiment, we chose a method that combines representations of multiple individually trained models that vary across the tasks used for representation learning. To extract the maximum possible information and make comparisons between the learned features, each model is trained using a single view. So, while some information may be redundant, using multiple independent models allows us to investigate the relationships between the learned transform tasks and our downstream tasks without introducing potential multi-task confounds. (See Fig 4.4.)
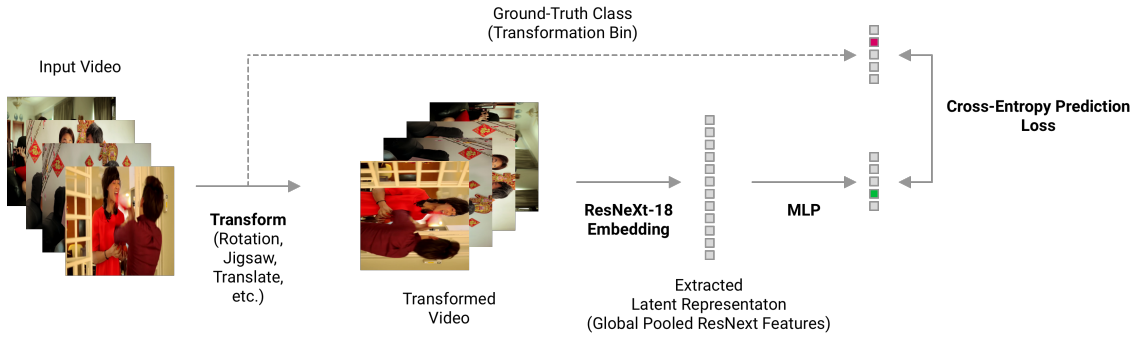
Figure 4.1: Overview of our pretext task algorithm. In the pretraining phase, a view transform as described in Section 3.1 is selected. We then use a ResNet based backbone to extract features from view realizations. These features are then passed through an MLP classification head which attempts to identity the view realization.
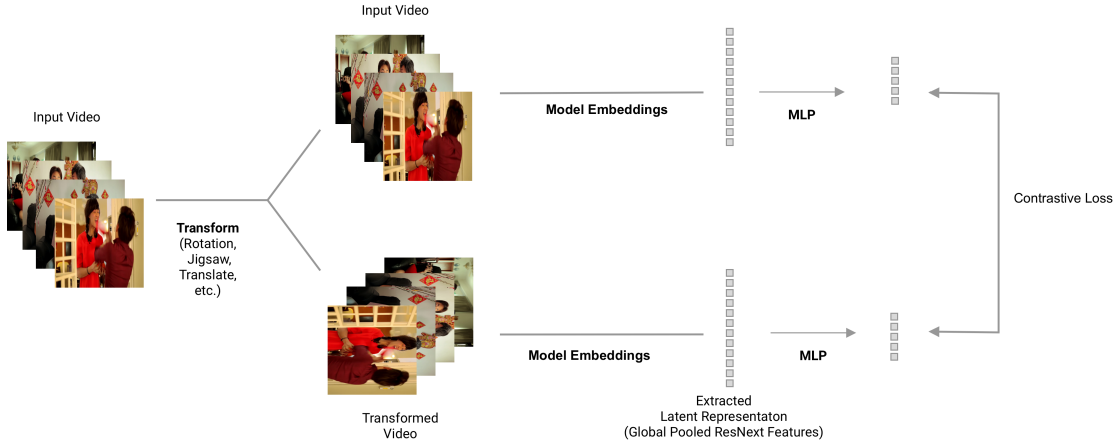
Figure 4.2: Overview of our contrastive learning algorithm. In the pretraining phase, a view transform as described in Section 3.1 is selected. These features are then passed through an MLP to a latent space. A contrastive loss is calculated using the two views and batch negative.
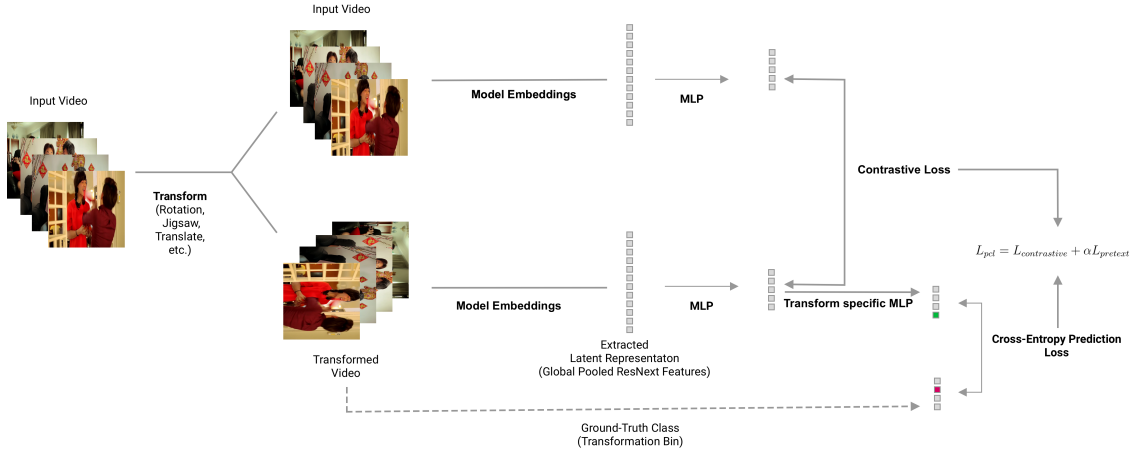
Figure 4.3: Overview of our pretext + contrastive learning algorithm. This algorithm combines the cross-entropy loss calculated in the pretext task with the contrastive loss computed during contrastive learning. α is a hyperparameter that balances the two losses.
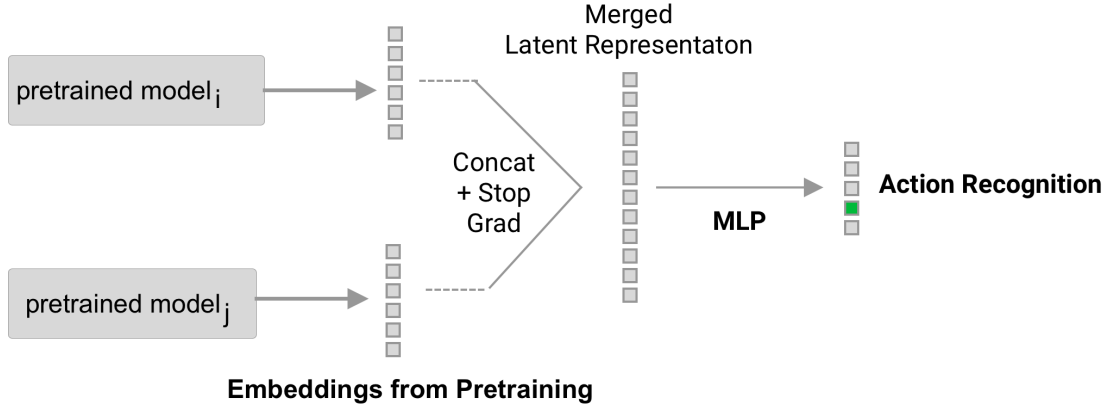
Figure 4.4: Overview of how we combine multiple independent tasks during finetuning. In the evaluation phase, we combine independently pretrained models by concatenating their embeddings and passing them to an MLP that predicts the action.

4.5 Datasets & Downstream Tasks

After selecting a view in the first phase, the two following phases of our experiments, pretraining and evaluation, rely on different video datasets. For pretraining, the model is trained on a large, unlabeled, video dataset and then it's evaluated on downstream video action recognition benchmarks.

For the second phase, all models in this work are pretrained on Kinetics-700 [10]. Kinetics-700 contains approximately 550,000 10-second training clips labeled from 1 of 700 action classes. We do not use these labels during pretraining.

For the third phase, we investigate the performance of these methods on video action recognition. We evaluated our models on UCF-101 [52] and HMDB [30]. UCF-101 is a video dataset containing 13,320 clips collected from YouTube representing 101 action categories. Clips range in length but can be up to 3 minutes long. We evaluate our methods using 3-fold evaluation, using the official THUMOS splits. Note that other methods show that training [27] on non-standard folds seems to improve performance, which we also saw in our experiments but refrain from discussing them in this work and instead focus on the official splits.

HMDB contains 6,800 clips collected from various sources, primarily movies, representing 51 actions. Videos in HMDB range in length but can be up to 30 minutes long. We evaluate our methods using 3-fold evaluation on the official provided splits (which each contain 70 train videos and 30 test videos).

4.6 Data Preprocessing

Techniques for preprocessing data can have varying impacts on downstream performance and requires further exploration in future work. For our preprocessing, each video is down-sampled to 4 FPS (an empirical trade-off which we found balances the ability to see high-frequency motion with data efficiency costs). We then change the video's spatial resolution to $(256, 256)$ by resizing the short edge of the video to length 256 and taking a center crop of the frames. The videos are then split temporally into sub-clips of 16 frames (4 seconds) which represent the inputs to our model. Note that we do not perform any mean-subtraction or whitening. To reiterate, we also do not perform any additional data augmentation during preprocessing for pretraining methods, as our goal is to evaluate the information captured by the pretraining method and introducing data augmentation could cause confounds in the downstream experiments. Data used for action recognition however is augmented using flipping to compare with some of the other SOTA video representation learning methods like in Jing *et al.* [25].

4.7 Model Architectures

Which model architectures work best for different pretraining tasks is still unknown. Kolesnikov *et al.* show that in pretext task-based training performance depends heavily on both the model architecture *and* pretext task being used [29]. Other more recent works have used transformer-based architectures to achieve SOTA results in image recognition [18], object detection [9], and video action recognition. Notably in the video domain, these architectures include ViViT, a pure-transformer based model using spatial and temporal heads in attention blocks, [5], TimeSformer which explores different self-attention schemes like divided space-time attention [7], CLIP [48] which uses transformer-based feature extractors on images and text (using ViT [18] for images), and VATT which also processes multimodal views using transformers [2]. Additionally, Lu *et al.* [34] use pretrained NLP transformers to achieve significant results in the vision domain. So, while these models may provide additional performance gains, we restrict ourselves to a single family of architectures, the ResNet family, to properly compare performance between pretraining experiments. We leave investigations into the ideal matching of views, learning algorithms, and models to future work. Note that following Harra et al. [21] we selected ResNeXt3D-18 as our backbone model, however, after we were unable to replicate results from the literature we experimented on ResNet3D-18, R(2+1)D, and ResNeXt3D-18 as a backbone for the Rotation 90 pretext task. Despite ResNeXt3D-18's superior performance on this though, we switched to this model backbone to make our results more comparable to the literature and reran pretext experiments using the ResNet3D-18 backbone. Results from this exploration are listed in Table: 5.2.

The ResNeXt3D-18 and ResNet3D-18 model consists of five convolutional layers where four of them contain two ResNeXt and ResNet blocks respectively. Each convolution uses 3D batch normalization and ReLU nonlinearities. A max-pooling layer follows the first con-

volutional layer, and adaptive average pooling follows the last layer, which unless otherwise specified compresses the last layer into a vector. The dimension of this vector v, is $v \in \mathbb{R}^{1024}$ unless otherwise stated.

Pretext task predictions are made by an MLP head on top of the ResNet backbone. This MLP head consists of 2 linear layers of dimension $\{1024, 512, C\}$ where C is the number of pretext classes. We use BatchNorm and ReLU nonlinearities between layers.

For contrastive tasks, latent representations of videos are extracted from MLPs feedforward networks that model sparse autoencoders, where the latent vector from the model is embedded into a larger dimension and then projected back down to the original dimension. LayerNorm, Dropout, and ReLU nonlinearities are used in the higher dimensional space. Two feedforward networks with latent dimensions 2048 are used and contrastive comparisons are made in $\mathbb{R}^{1024}$.

4.8   Experiments & Implementation details

In this section, we quantify the performance of different pretraining algorithms using different view transforms by analyzing transfer performance on video action recognition benchmarks. Note that to effectively analyze the learned representations we **completely freeze the feature extraction model** and don't initialize an unfrozen model that's trained in a fully supervised manner. We speculate that performance could be improved if the extractor was fully trainable but we leave this for future work. For all experiments, we use PyTorch [43] and PyTorchLightning [19] and we train on Google Cloud Platform with four Tesla T4 GPUs where each experiment uses a single GPU. We also use open source packages like NumPy [22], Scikit-Image [46], PIL [15] and OpenCV [8].

Pretraining

For all *pretext experiments* we use the Adam optimizer with a learning rate of 3e-4 that decays according to a fixed multi-step learning rate schedule of .1 after 80 and 100 epochs with an effective batch size of 1024. The model for pretext task experiments is optimized using a cross-entropy loss where all pretext tasks require predicting which realizations of the view transform was used. For all *contrastive experiments*, we change the learning rate scheduler to a cosine learning rate scheduler that oscillates between 1e-7 and 1e-4 every 8000 steps. We also utilize the Normalized Temperature-scaled Cross Entropy (NT-Xent) contrastive loss from SimCLR [12] with the temperature parameter set to 1.0. The batch size for contrastive learning is 64 as this was the maximum batch size that fit in memory given our setup. *Pretext + contrastive learning* follows the same optimization and learning rate schedule as contrastive learning. The loss of pretext + contrastive is a combination of both the pretext losses and the contrastive losses as described in 4.2. For pretext + contrastive training, α is set to 10 and the batch size is 55 (again the maximum size that fit in memory given our setup).

Finetuning on Action Recognition

Fintetuning experiments are performed on the action recognition task. We evaluate the classification performance using a MLP head with four linear layers of dimension $2048, 1048, 512, C_{task}$ where each layer uses BatchNorm1D and a HardSwish [49] nonlinear function. The MLP is optimized using standard cross-entropy loss and the Adam optimizer with learning rate $3e$-4 that decays by .1 according to a fixed multi-step learning rate scheduler of 20 and 80 epochs. To compare with other SOTA methods like Jing *et al.* [25] we additionally augment the data during finetuning with a horizontal flip augmentation (even though this is avoided during pretraining). For HMDB, only to prevent overfitting on the relatively small training set, we employ a dropout after each non-linearity. We also experimented with logistic regression probes using the standard logistic regression module available in Scikit Learn [46] but unfortunately, the results were inconclusive.

Combination-of-task experiments follow the same setup as pretext experiments except that the MLP head input dimension is expanded to accommodate the number of combined features which varies depending on the number of embeddings combined. Model backbones will be explicitly specified.

Chapter 5

Results and Analysis

Our main results are presented in Tables: 5.3, 5.5 and 5.16. Rotation 90° is our best single view transform for pretext based methods and outperforms supervised baselines on UCF-101 and HMDB while for contrastive methods Rotation 90° underperforms them. Further work investigating the model's performance on UCF-101 and HMDB is required as we note a disparity between our scores and some scores reported in the literature, notably that of Jing *et al.* [25]. Note that all reported accuracies are clip accuracies unless otherwise stated.

5.1   Pretext Task Experiments

Single-Transform Experiments

In this section, we present detailed results of the single view transform pretext task pretraining experiments. Table 5.2 reports validation accuracies of pretext scores. Note that in many cases, higher pretraining accuracy seems to denote a simpler learning task and may perform worse on downstream tasks. Table 5.3 presents performance for a single pretext task on UCF-101 and HMDB. The MLP scores are the average of all available splits. As a baseline, these scores are compared to the accuracies of purely supervised training regiments, where for each train/test split of the action recognition datasets, a model trains using supervised learning and is then evaluated on the test split. Scores are averages across splits unless otherwise specified.

As mentioned earlier in Section 4.1, the goal of pretext task learning is to train a model that extracts information useful for the pretext and a downstream task. We hypothesize that the best models extract information that separates according to challenging and dissimilar transform realizations in the latent space.

We speculate that rotation is one of the more challenging tasks to learn to separate, as it requires models to learn about the standard orientation of objects and locally, at the scale of convolutions, this can be challenging. Difficult examples to classify are ones with radial symmetry which may be infrequent when compared to difficult examples under other

Model Config.	Pretext Val. Accuracy (MLP)	UCF-101 (MLP)	HMDB (MLP)
ResNeXt3D-18	96.37%	43.43%	25.54%
ResNet3D-18	93.75 %	42.98%	22.49 %
ResNet(2+1)D-18	87.27%	31.16%	15.54%

Table 5.1: Pretext Training top-1 MLP clip accuracy on UCF-101 and HMDB after pretext task training on Rotation 90.

transforms such as flips or translations which occur more frequently. On the other hand, view transforms which perform poorly such as the Color Jitter transform seem to share little information with the final downstream task - indeed, learning relative pixel magnitudes may not be all that useful when determining actions but may be better suited to other downstream tasks where color information is more relevant like video segmentation.

It's interesting to note that temporal view transforms such as pace and temporal jigsaw do not do well on their own compared to other pretext methods. We hypothesize that while these temporal views allow the model to learn important temporal features the information is not as helpful as the learned spatial information from spatial views. The results from Radford *et al.* [48] support this hypothesis as SOTA performance on UCF-101 only uses a single frame for action recognition and completely ignores temporal information.

5.2   Model Exploration

As described in 4.7, following the results of 5.1, we were bothered by the inability to reproduce the scores from Jing *et al.* [25] on UCF-101, despite using higher resolution videos as well as a model structure that we hypothesized would more effective than 3DRotNet's ResNet3D-18 backbone. For reference, 3DRotNet reached 62.8%, 32.5% accuracy on UCF-101 and HMDB respectively while we reached 43.43%, 25.54 using a ResNeXt3D-18. We were able to see similar performance gains relative to a supervised baseline like Jing *et al.*' on both action datasets. There has been little research into the optimal preprocessing methods for video representation learning and possible differences may be due to our optimization methods (using an Adam optimizer instead of an SGD optimizer, learning rate scheduler with milestones) which could be worse for all pretraining tasks.

Additionally, these results indicate high variance in action recognition across backbones, even if the backbones are from the same family. As discussed in Section: 4.7, video action recognition scores vary significantly across models despite restricting the set of models to the same family. These results recapitulate the ideas from Kolesnikov *et al.* [29] that for each learning framework the optimal model requires exploring a plethora of different architectures.

5.3 Contrastive Task Experiments

Single-Transform Experiments

In this section we present detailed results of the single view transform contrastive experiments, selecting view transforms that try to extend the work by Chen *et al.* [12] for video action recognition. We note however that Chen *et al.* [12] explored combinations of transforms by composing them while we purposefully study single view transforms in isolation. Table 5.4 outlines the pretraining performance of single view transforms. Table 5.5 details the MLP scores on action recognition datasets. All scores are the average of all available splits unless otherwise stated. As a baseline, these scores are compared to the accuracies of purely supervised methods.

The most surprising result was the correlation between best-performing views in Sim-CLR's ImageNet classification task [12] and worst performing views for our action recognition task. Cutout, which achieved the second-best single view accuracy in SimCLR [12], performed the worst on the action recognition task. Note though, that the poor performance of rotation seems to mirror the performance in SimCLR. Our new proposed view transform, FFT, performs the best in the contrastive learning setup, achieving performance that nearly matches supervised learning. Temporal views also seem to outperform their spatial counterpart unlike in pretext task training. Temporal views may conform to the conditions of a good view presented by Tian *et al.* [55] since they share less information between views than spatial views. This idea is supported by spatiotemporal view transforms performing nearly as well as purely temporal view transforms despite the additional changes to the spatial dimension.

5.4 Combination of Tasks

For the majority of tasks, a single pretraining algorithm is not sufficient to beat the performance of supervised learning benchmarks, except for Rotation 90° as show in in Tables: 5.3, 5.5. By combining several of these transforms however we were generally able to outperform our supervised baselines and we notice that as we added more pretext tasks we were able to improve performance while for contrastive learning this wasn't always the case.

Pretext combinations only

Focusing on pretext task pretrained models, we present expanded results when using combinations of several transforms in Tables: 5.6, 5.7, 5.8, 5.9, 5.10, 5.11, 5.12. Learned embeddings are combined using the methods described in Section: 4.4. We also explore different features learned by different rotation tasks to see how modifying the complexity of the view transform alters the learned features. We note in Table: 5.10 that despite using the same view transform, modifying the rotation appears to learn orthogonal features as performance

improves when multiple rotations are combined. Generally, we see that combinations of model embeddings trained on different pretext tasks appear to do better than either single pretext task. This seems reasonable as adding additional representations increases the amount of information captured in the features. The best combination of pretext tasks using the ResNeXt3D-18 visual backbone comes from Rotation 90° and Rotation 12° on UCF101 and Rotation 90° and Rotation 12° on HMDB. It appears that this trend does not increase as more embeddings are combined. It's possible that beyond a certain latent dimension size the MLP may contain too many parameters and be overfitting to the downstream training split during finetuning.

Contrastive combinations only

Combining contrastive transforms interestingly does not always mirror the performance gains seen in pretext combinations. We present expanded results when using combinations of several transforms in Tables: 5.13, 5.14, 5.15, and 5.15.

While most combinations increase in performance when combined, like the temporal views, spatiotemporal views, and FFT views, others like Cutout when combined with FFT perform *worse* than a single independent model. This suggests that the MLPs are unable to reconcile the different embeddings from the different models. It's possible that during the training phase, the model considers both features equally and gets trapped in a local minimum that uses both features rather than completely ignore the feature from the Cutout pretrained model resulting in performance that's closer to the average of the two. While we experimented with other optimization routines and learning rate schedulers we were unable to reconcile this unintuitive decrease in performance.

Pretext + Contrastive

In this section, we present results on our preliminary investigations on pretext + contrastive learning by analyzing this framework using Rotation 90° and FFT which performed optimally on the action recognition task when transferred from pretext task training and contrastive learning. We also compare this method to a combination task that uses Rotation 90° and FFT views. Results are presented in 5.16

The most startling preliminary result from these experiments is that pretext + contrastive learning significantly underperforms our best model Rotation 90°. This may be due to a model that has not converged yet as this model trained for less than 100 epochs. Another interesting observation is that the Rotation 90° and FFT combination task performs much better than the pretext + contrastive learning task. Clearly, more work is necessary to fully analyze this pretraining algorithm.

Comparison to other methods

Table: 5.17 gives the performance of other SOTA self-supervised models. In many cases, our methods appear to perform better than supervised baselines in our setup but significantly underperform results reported in the literature. On the UCF-101 dataset, our best models fail to outperform transfer from a pretrained Kinetics model. This is not surprising as pretext tasks with objectives that align closer to downstream task objects are best suited to capture downstream relevant information. Thus a model that trains on a more difficult action recognition task contains features that transfer easily to another action recognition task.

Another core difference between our methods and other work is that we fully freeze our feature extractor. Some experiments performed by Jing *et al.* explicitly test this [25] by unfreezing different levels of the network and observing better performance compared to a fully frozen network.

More work is needed however to reconcile what exactly is causing these disparities in scores.

Transform (s)	Transform Config	Pretext Val. Accuracy (MLP)
AutoContrast	on-off, 2 classes	79.97%
Color Jitter	$[.1, 1.1]^3 \times [-.5, .5]$, $4 \times 4 \times 4 \times 5$ multilabel classes,	22.20%
Edge Filter	10 edge filters, 10 classes	83.75%
FFT (Circular Mask)	14 frequency masks	100. %
Flip	left, right, up, down, 4 classes	43.75%
Invert Order	on-off, 2 classes	50.03%
Pace	4 paces, 4 classes	76.84%
Rotation	90°, 4 classes	96.37%
Rotation	72°, 5 classes,	99.93%
Rotation	60°, 6 classes	89.41%
Rotation	30°, 12 classes	80.29%
Rotation+Zoom	72°, 5 classes	99.28%
Shear X&Y	$[-.5, .5]^2$, 16 classes	99.15%
Spatial Jigsaw (Permutation)	4 pieces, 24 classes	96.86%
Spatial Jigsaw (Relative)	4 pieces, 4^4 multilabel classes	71.94%
Spatial Jigsaw (Relative)	9 pieces, 9^9 multilabel classes	49.23%
Temporal Jigsaw	4 pieces, 24 classes	52.44%
Translate X&Y	$[-150, 150]^2$, 36 classes	98.03%
Translate+Zoom	10 scales, 10 classes	43.92%
Zoom	10 scales, 10 classes	63.99%
3DRotNet [25]	90°, 4 classes	92.72%
Video Cloze Procedure (VCP) [35]	5 spatial, temporal transform, 5 classes	78.42%

Table 5.2: Pretext Training top-1 pretraining accuracy on Kinetics-700. ResNeXt3D-18 is used as the backbone for all view transforms.

Pretext Transform	UCF-101 (MLP)	HMDB (MLP)
Supervised Training	*35.61%*	*14.13%*
Flip	27.59% ± 0.58%	14.49% ±1.48%
Rotation-90	**43.43** % ± 1.16%	**25.54** % ±2.83%
Rotation-90 ResNet3D-18	42.98%	22.49%
Rotation-72	13.15% ± 1.91%	6.55% ±0.08%
Rotation-60	33.29% ± 0.10%	19.27% ±0.24%
Rotation-12	28.85% ± 0.19%	14.39% ±0.12%
Shear-0.25	11.53% ± 1.54%	12.19% ±11.47%
Translate+Zoom-9	14.17% ± 0.44%	7.14% ±0.06%
Spatial Jigsaw (Permutation)-4	23.29 % ± 0.49%	10.89% ± 0.25%
Spatial Jigsaw (Relative)-9	30.37% ± 0.63%	16.00% ±0.92%
Spatial Jigsaw (Relative)-4 ResNet3D-18	26.67%	12.83%
Spatial Jigsaw (Relative)-4	14.63% ± 0.64%	7.78% ±0.36%
Pace-$\{.5, 1, 2, 3, 4\}$	16.27% ± 0.01%	7.89% ±0.02%
AutoContrast	13.92% ± 0.27%	7.88% ±0.08%
Color Jitter	14.59% ± 0.78%	7.48% ±0.54%
Edge-Filter-10	7.71% ± 0.47%	5.88% ±1.62%
Zoom-9	9.79% ± 1.65%	8.02% ±0.09%
Temporal Jigsaw-4	14.88% ± 0.08%	8.25% ±0.11%
Rotate+Zoom-72	16.24% ± 0.45%	9.05% ± 0.43%
Invert Order	5.420%	3.717%
Translate	6.80%	1.524%
FFT (Circular Mask) ResNet3D-18	27.33 %	35.00%

Table 5.3: Top-1 action recognition accuracy on UCF-101 and HMDB when using a single pretext task for training. The rotation methods perform optimally, with decreased performance when using five discretization buckets. These tasks are followed in performance by the Spatial Jigsaw, Flip and Autocontrast methods. All methods use a ResNeXt3D-18 backbone unless explicitly mentioned in the config.

Transform (s)	Transform Config	Contrastive Loss	Val. Accuracy (MLP)
Cutout	6 crops scales	7.421	83.63%
FFT (Circular Mask)	14 masks	7.811	97.15%
FFT (Torus Mask)	12 masks	7.812	99.52%
Rotation	90°	7.731	99.98%
Temporal Jigsaw+Pace	4 temporal chunks x 4 paces	7.791	99.37%
Spatiotemporal Jigsaw	4x4x4 spatiotemporal chunks	7.696	99.98%

Table 5.4: Contrastive Training top-1 pretraining accuracy on Kinetics-700. Models used for all transforms are ResNeXt3D-18

Contrastive Transform	UCF-101 (MLP)	HMDB (MLP)
Supervised Training	*36.36%**	*26.00%**
Cutout	9.184%	4.702%
FFT (Circular Mask)	33.19%	17.41%
FFT (Torus Mask)	**31.43** %	**16.60** %
Rotation-90	13.25 %	9.937%
Temporal Jigsaw+Pace	24.85 %	15.27 %
Spatiotemporal Jigsaw	24.81 %	12.11 %

Table 5.5: Top-1 action recognition accuracy on UCF-101 and HMDB when using a single transform for contrastive learning. All methods use a ResNet3D-18 backbone. A "*" indicates experiments run only on the first split.

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP)
Pace-$\{.5, 1, 2, 3, 4\}$, Rotation-90	**46.77%**	**25.55%**
Pace-$\{.5, 1, 2, 3, 4\}$, Rotation-72	19.56%	9.400%
Pace-$\{.5, 1, 2, 3, 4\}$, Rotation-60	36.61%	20.34%
Pace-$\{.5, 1, 2, 3, 4\}$, Rotation-12	30.88%	16.41%
Pace-$\{.5, 1, 2, 3, 4\}$, Spatial Jigsaw (permutation)-4	27.69%	13.09%
Pace-$\{.5, 1, 2, 3, 4\}$, Spatial Jigsaw (relative)-9	31.72%	18.63%
Pace-$\{.5, 1, 2, 3, 4\}$, Shear X,Y-.25	18.05 %	8.076%
Pace-$\{.5, 1, 2, 3, 4\}$, Translate-50	19.99%	9.605%
Pace-$\{.5, 1, 2, 3, 4\}$, Translate+Zoom-9	19.31%	10.36%

Table 5.6: Action Recognition top-1 accuracies on all datasets when using a pretext-task paired with the Pace temporal transform. All methods use a resolution of (16,256,256,3) and the ResNeXt-18 visual backbone. UCF-101 (MLP) split values are reported on the first split only.

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP)
Auto Contrast, Color Jitter-.25	19.23 %	10.14%
Auto Contrast, Edge Filter-10	14.97%	7.896%
Auto Contrast, Flip	28.32%	14.60%
Auto Contrast, Rotation-90	**43.21%**	**22.75 %**
Auto Contrast, Rotation-72	16.74%	8.898%
Auto Contrast, Rotation-60	33.69%	18.17%
Auto Contrast, Rotation-12	28.76%	14.32%
Auto Contrast, Spatial Jigsaw (permutation,4)	23.92%	11.88%
Auto Contrast, Spatial Jigsaw (relative,9)	30.73%	18.14%
Auto Contrast, Pace-$\{.5, 1, 2, 3, 4\}$	20.3%	11.51%
Auto Contrast, Shear X,Y-.25	14.81%	8.364%
Auto Contrast, Translate-50	17.11%	8.819%

Table 5.7: Action Recognition top-1 accuracies on all datasets when using a pretext-task paired with Auto Contrast. All methods use a resolution of (16,256,256,3) and the ResNeXt-18 visual backbone. UCF-101 reports accuracies on the first split only in order to compare to 3DRotNet [25]. HMDB mean scores on all three splits are reported

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP)
Color Jitter, EdgeFilter-10	8.540%	7.581%
Color Jitter, Flip	28.38%	15.58%
Color Jitter, Rotation-90	**43.27%**	15.02%
Color Jitter, Rotation-72	17.7%	9.073%
Color Jitter, Rotation-60	34.63%	**17.06%**
Color Jitter, Rotation-12	29.16%	15.02%
Color Jitter, Spatial Jigsaw-4	25.51%	11.43%
Color Jitter, Spatial Jigsaw (relative) -9	30.64%	15.83%
Color Jitter, Shear X,Y -.25	16.2%	7.910%
Color Jitter, Translate-50	17.87%	8.052%
Color Jitter, Translate+Zoom-9	19.34%	10.70%

Table 5.8: Action Recognition top-1 accuracies on all datasets when using a pretext-task paired with ColorJitter. All methods use a resolution of (16,256,256,3) and the ResNeXt-18 visual backbone.

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP)
Edge Filter-10, Flip	14.89%	14.56%
Edge Filter-10, Rotation-90	44.44%	24.05%
Edge Filter-10, Rotation-72	13.98%	7.42%
Edge Filter-10, Rotation-60	32.36%	17.83%
Edge Filter-10, Rotation-12	29.94%	13.90%
Edge Filter-10, SpatialJigsaw (permutation)-4	22.57%	10.23%
Edge Filter-10, SpatialJigsaw (relative)-9	31.13%	16.89%
Edge Filter-10, Shear X,Y -.25	11.95%	5.887%
Edge Filter-10, Translate -50	12.89%	8.15%
Edge Filter-10, Translate+Zoom	15.04%	8.15%
Flip, Rotation-90	**45.87%**	**24.76%**
Flip, Rotation-72	28.94%	15.44%
Flip, Rotation-60	35.61%	19.56%
Flip, Rotation-12	31.33%	16.74%
Flip, SpatialJigsaw (permutation)-4	31.5%	15.46%
Flip, SpatialJigsaw (relative) -9	37.33%	20.83%
Flip, Shear X,Y -.25	28.89%	15.48%
Flip, Translate-50	28.52%	15.66%
Flip, Translate+Zoom	29.52%	15.48%

Table 5.9: Action Recognition top-1 accuracies on all datasets when using a pretext-task paired with Edge Filter-10 or Flip. All methods use a resolution of (16,256,256,3) and the ResNeXt-18 visual backbone.

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP)
Rotation-90, Rotation-72	44.24%	22.65%
Rotation-90, Rotation-60	**50.05%**	**25.11%**
Rotation-90, Rotation-12	47.03%	23.77%
Rotation-72, Rotation-60	34.07%	18.31%
Rotation-72, Rotation-12	80.39%	14.22%
Rotation-60, Rotation-12	36.78%	19.79%

Table 5.10: Action Recognition top-1 accuracies combinations of different rotations. All methods use a resolution of (16,256,256,3) and the ResNeXt-18 visual backbone.

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP)
Rotation-90, Spatial Jigsaw (permutation)-4	**45.36**%	23.29%
Rotation-90, Shear X,Y -.25	44.31%	23.21%
Rotation-90, Translate+Zoom-9	44.38%	**23.61**%

Table 5.11: Action Recognition top-1 accuracies on all datasets when using a pretext-task paired with the Rotation-task. All methods use a resolution of (16,256,256,3) and the ResNeXt-18 visual backbone.

Pretext Transform (s)	UCF-101 (MLP) split-1	HMDB (MLP) split-1
Rotation-90,Rotation-72,Rotation-60,Rotation-12,Flip, Shear-.25, Spatial Jigsaw (Permutation,4), Spatial Jigsaw (Relative)-4, Spatial Jigsaw (relative)-9, Translate-50, Translate+Zoom-10,Pace-$\{.5, 1, 2, 3, 4\}$, Invert Order, Temporal Jigsaw-4,Autocontrast, Color Jitter-.25, Edge-10	46.25%	19.27%

Table 5.12: Action Recognition top-1 accuracy on the first split in UCF101 and HMDB when models trained on pretext tasks using all spatial, all temporal, and all visual transforms combined. The MLP from Section 4 is modified to $\{10000, 5000, 2048, 1024, 512, C\}$ to accommodate the large feature space

View Transform (s)	UCF-101 (MLP)	HMDB (MLP)
Cutout, FFT (Circular Mask)	28.5%	**11.59**%
Cutout, FFT (Torus Mask)	**29.65**%	8.353%
Cutout, Rotate	17.79%	7.698%
Cutout, Temporal Jigsaw+Pace	25.79%	10.81%
Cutout, Spatiotemporal Jigsaw	22.02%	9.789%

Table 5.13: Action Recognition top-1 accuracies on all datasets when combining transforms with the Cutout embeddings from contrastive learning. All methods use a resolution of (16,256,256,3) and the ResNet-18 visual backbone.

View Transform (s)	UCF-101 (MLP)	HMDB (MLP)
FFT (Circular Mask), FFT (Torus Mask)	33.27 %	10.82 %
FFT (Circular Mask), Rotate	24.72%	9.534 %
FFT (Circular Mask), Temporal Jigsaw+Pace	34.65%	10.78%
FFT (Circular Mask), Spatiotemporal Jigsaw	29.08%	10.29%
FFT (Torus Mask), Rotate	25.96%	12.87%
FFT (Torus Mask), Temporal Jigsaw+Pace	**35.9** %	**15.92**%
FFT (Torus Mask), Spatiotemporal Jigsaw	30.66%	14.45%

Table 5.14: Action Recognition top-1 accuracies on all datasets when combining transforms with the FFT embeddings from contrastive learning. All methods use a resolution of (16,256,256,3) and the ResNet-18 visual backbone.

View Transform (s)	UCF-101 (MLP)	HMDB (MLP)
Rotate-90, Temporal Jigsaw+Pace	23.81%	11.11%
Rotate-90, Spatiotemporal Jigsaw	19.75%	9.412%
Temporal Jigsaw+Pace, Spatiotemporal Jigsaw	**26.49**%	**11.94**%

Table 5.15: Action Recognition top-1 accuracies on all datasets when combining transforms with the Rotate 90 and Temporal Jigsaw+Pace embeddings from contrastive learning. All methods use a resolution of (16,256,256,3) and the ResNet-18 visual backbone.

View Transforms	Method	UCF101 (MLP)	HMDB (MLP)
CircularFFT, Rotation-90	pretext+contrastive learning	32.38%	**29.42%**
CircularFFT, Rotation-90	pretext, contrastive learning combined embeddings	**46.42** %	23.29 %

Table 5.16: Top-1 action recognition accuracy on UCF-101 and HMDB when using a pretext+contrastive learning for our two best view transforms. We also compare this to combining their independently learned embeddings from pretext task learning and contrastive learning. All methods use a ResNet3D-18 backbone

Pretrained on	Method	Resolution	Model	UCF101	HMDB
Kinetics-400	STPuzzle [28]	224	ResNet3D-18	65.8%	33.7%
Kinetics-400	VCOP [67]	112	R(2+1)D	72.4%	30.9%
Kinetics-400	VRLPP [59]	112	R(2+1)D-18	75.9%	35.9%
Kinetics-400	VTHCL [68]	112	ResNet3D-18	80.6%	48.6%
Kinetics-400	PRP [69]	112	ResNet3D18	66.5%	29.7%
Kinetics-400	STS [60]	112	R(2+1)D-18	77.8%	40.7%
Kinetics-400	Lorre *et al* [33]	224	ResNet-18	70.5%	41.1%
Kinetics-400	RSPNet [11]	112	ResNet3D-18	74.3%	41.8%
Kinetics-400	IDT [56]	112	ResNet3D-18	73%	41.6%
Kinetics-400	SSTL [61]	112	ResNet3D-18	79.1%	49.7%
Kinetics-400	TaCo [6]	112	ResNet3D-18	85.1%	51.6%
	TaCo Frozen	112	ResNet3D-18	59.63	26.7
Kinetics-400	DPC [20]	128	ResNet3D-18	68.2%	34.5%
Kinetics-400	VIE [70]	112	ResNet3D-18	72.3%	44.8%
Kinetics-400	Supervised pretraining [26]	112	ResNeXt101	97.46%	81.79%
Kinetics-400	ResNeXt101 BERT [26]	112	ResNeXt101 BERT	98.10%	83.55%
Kinetics-600	Temporal Transform Prediction (RTT) [24]	112	ResNet3D-18	79.3%	49.8%
Kinetics-600	CVRL [47]	112	ResNet3D-50	92.2%	66.7%
Kinetics-700	Rotation-90 (ours)	256	ResNeXt-18	43.43%	25.54%
Kinetics-700	Rotation-90, Rotation-60 (ours)	256	ResNeXt-18	50.05%	25.11%
Kinetics-700	Circular FFT(ours)	256	ResNet3D-18	31.43 %	16.60 %
Kinetics-700	Circular FFT, Temporal Jigsaw+Pace(ours)	256	ResNet3D-18	35.9%	15.92%
Kinetics-700	VCP [35]	112	C3D	68.5%	32.5%
Kinetics-700	OPN [31]	112	ResNet3D-18	56.3%	22.1%
Kinetics-700	Pace Pred [14]	112	R(2+1)D-18	74.82%	36.82%
Kinetics-700	TCL [16]	112	ResNet3D-18	84.1%	53.6%
Kinetics-700	VideoMoCo [42]	112	ResNet3D-18	74.1%	43.6%
Kinetics-700	3DRotNet [25]	112	ResNet3D-18	62.8%	29.6%
Kinetics-700	Jigsaw [1]	112	CaffeNet	55.4%	27%
Kinetics-700	Pretext+Contrastive [54]	112	ResNet3D-18	82.3%	43.2%
Kinetics-700	Supervised pretraining [27]	112	ResNet3D-18	87.9%	57.1%
ASR	VATT [2]	224	Transformers	89.6%	65.2%
WIT	CLIP + LR [48]	112	ResNet50-2D	76.4%	-
Kinetics-700	HAF+BoW/FV Hallucinating [62]	112	I3D	-	82.37%
IG65M	R(2+1)D BERT [26]	112	R(2+1)D BERT	**98.96%**	**85.1%**

Table 5.17: Comparison of our methods to state of the art methods on video action recognition in UCF-101 and HMDB

Chapter 6

Future Work

This work gives an overview of the different performances of view transforms in combination with current SOTA video representation learning algorithms, however further work is required to fully understand some emerging questions in this study and this field. Unintuitive combinations of learning algorithms, like pretext + contrastive, perform well on downstream video tasks and illustrates that we do not fully understand the kind of representations that are being extracted by the model. It also suggests that combining multiple pretraining experiments could be helpful in practice and one interesting study could incorporate multimodal learning with pretext and contrastive learning. Another important area that needs to be explored further is whether or not certain kinds of view transforms perform well with certain architectures; while the ResNet model family was selected as a control for this experiment other architectures like Transformers should be investigated. These self-supervised learning tasks can also be considered as a video-to-video translation task where the view transforms are simply translations between videos. With this in mind, we should explore SOTA translation architectures, like image-to-image style GANs and expand on techniques like the ones proposed by Pan *et al.* [42] that incorporates adversarial examples. Another line of research that would be interesting to investigate following recent successes in Bertasius *et al.* [7] (who introduce large amounts of inductive bias to a transformer during training), would be to explore the effects of explicitly biasing the latent space. Such biases could explicitly ask models trained to enforce some sort of structure during pretext training or contrastive training, which would also have the added benefit of making the space more interpretable. Finally, and most importantly, all combinations of view transform, architecture, and video tasks should be compared to accurately determine whether there is an underlying pattern that boosts performance in all pretraining techniques.

Chapter 7

Conclusion

In this work, we explored multiple view transforms and their interactions with pretext task learning and contrastive learning on the video action recognition task. We introduce new view transforms, FFT-based spatial filters, and a new method, the combination task, for analyzing individual pretraining tasks by combining learned embeddings that are then transferred to a downstream task. We observe that the Rotation 90° pretext task pretraining transfers the best to video action recognition benchmarks while Rotation 90° contrastive pretraining does poorly. We also observe that FFT-based spatial filter contrastive pretraing transfers the best for all contrastive learning methods. Additionally, in our combination task experiments, combining features generally boosts performance on video action recognition benchmarks, implying that different view transforms with different pretraining techniques capture orthogonal features. We are encouraged by our findings however more work is necessary to fully understand what information is extracted by self-supervised learning techniques and make our results more comparable to the literature.

Bibliography

[1]   Unaiza Ahsan, Rishi Madhok, and Irfan A. Essa. "Video Jigsaw: Unsupervised Learning of Spatiotemporal Context for Video Action Recognition". In: *CoRR* abs/1808.07507 (2018). arXiv: 1808.07507. URL: http://arxiv.org/abs/1808.07507.

[2]   Hassan Akbari et al. *VATT: Transformers for Multimodal Self-Supervised Learning from Raw Video, Audio and Text.* 2021. arXiv: 2104.11178 [cs.CV].

[3]   Humam Alwassel et al. *Self-Supervised Learning by Cross-Modal Audio-Video Clustering.* 2020. arXiv: 1911.12667 [cs.CV].

[4]   Relja Arandjelović and Andrew Zisserman. *Look, Listen and Learn.* 2017. arXiv: 1705.08168 [cs.CV].

[5]   Anurag Arnab et al. *ViViT: A Video Vision Transformer.* 2021. arXiv: 2103.15691 [cs.CV].

[6]   Yutong Bai et al. *Can Temporal Information Help with Contrastive Self-Supervised Learning?* 2020. arXiv: 2011.13046 [cs.CV].

[7]   Gedas Bertasius, Heng Wang, and Lorenzo Torresani. *Is Space-Time Attention All You Need for Video Understanding?* 2021. arXiv: 2102.05095 [cs.CV].

[8]   G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[9]   Nicolas Carion et al. *End-to-End Object Detection with Transformers.* 2020. arXiv: 2005.12872 [cs.CV].

[10]   Joao Carreira et al. "A short note on the kinetics-700 human action dataset". In: *arXiv preprint arXiv:1907.06987* (2019).

[11]   Peihao Chen et al. *RSPNet: Relative Speed Perception for Unsupervised Video Representation Learning.* 2021. arXiv: 2011.07949 [cs.CV].

[12]   Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations.* 2020. arXiv: 2002.05709 [cs.LG].

[13]   Ting Chen et al. *Big Self-Supervised Models are Strong Semi-Supervised Learners.* 2020. arXiv: 2006.10029 [cs.LG].

[14]   Hyeon Cho et al. "Self-Supervised Spatio-Temporal Representation Learning Using Variable Playback Speed Prediction". In: *CoRR* abs/2003.02692 (2020). arXiv: 2003.02692. URL: https://arxiv.org/abs/2003.02692.

[15] Alex Clark. *Pillow (PIL Fork) Documentation*. 2015. URL: https://buildmedia. readthedocs.org/media/pdf/pillow/latest/pillow.pdf.

[16] Ishan Dave et al. *TCLR: Temporal Contrastive Learning for Video Representation*. 2021. arXiv: 2101.07974 [cs.CV].

[17] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. "Unsupervised Visual Representation Learning by Context Prediction". In: *CoRR* abs/1505.05192 (2015). arXiv: 1505.05192. URL: http://arxiv.org/abs/1505.05192.

[18] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. arXiv: 2010.11929 [cs.CV].

[19] et al. Falcon WA. "PyTorch Lightning". In: *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning* 3 (2019).

[20] Tengda Han, Weidi Xie, and Andrew Zisserman. *Video Representation Learning by Dense Predictive Coding*. 2019. arXiv: 1909.04656 [cs.CV].

[21] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. "Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?" In: *CoRR* abs/1711.09577 (2017). arXiv: 1711.09577. URL: http://arxiv.org/abs/1711.09577.

[22] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10. 1038/s41586-020-2649-2.

[23] Yuqi Huo et al. *Self-Supervised Video Representation Learning with Constrained Spatiotemporal Jigsaw*. 2021. URL: https://openreview.net/forum?id=4AWko4A35ss.

[24] Simon Jenni, Givi Meishvili, and Paolo Favaro. *Video Representation Learning by Recognizing Temporal Transformations*. 2020. arXiv: 2007.10730 [cs.CV].

[25] Longlong Jing and Yingli Tian. "Self-supervised Spatiotemporal Feature Learning by Video Geometric Transformations". In: *CoRR* abs/1811.11387 (2018). arXiv: 1811. 11387. URL: http://arxiv.org/abs/1811.11387.

[26] M. Esat Kalfaoglu, Sinan Kalkan, and A. Aydin Alatan. *Late Temporal Modeling in 3D CNN Architectures with BERT for Action Recognition*. 2020. arXiv: 2008.01232 [cs.CV].

[27] Hirokatsu Kataoka et al. *Would Mega-scale Datasets Further Enhance Spatiotemporal 3D CNNs?* 2020. arXiv: 2004.04968 [cs.CV].

[28] Dahun Kim, Donghyeon Cho, and In So Kweon. "Self-Supervised Video Representation Learning with Space-Time Cubic Puzzles". In: *CoRR* abs/1811.09795 (2018). arXiv: 1811.09795. URL: http://arxiv.org/abs/1811.09795.

[29] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. "Revisiting Self-Supervised Visual Representation Learning". In: *CoRR* abs/1901.09005 (2019). arXiv: 1901.09005. URL: http://arxiv.org/abs/1901.09005.

[30] Hildegard Kuehne et al. "HMDB: a large video database for human motion recognition". In: *2011 International conference on computer vision.* IEEE. 2011, pp. 2556–2563.

[31] Hsin-Ying Lee et al. *Unsupervised Representation Learning by Sorting Sequences.* 2017. arXiv: 1708.01246 [cs.CV].

[32] Junnan Li et al. *Prototypical Contrastive Learning of Unsupervised Representations.* 2021. arXiv: 2005.04966 [cs.CV].

[33] Guillaume LORRE et al. "Temporal Contrastive Pretraining for Video Action Recognition". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV).* Mar. 2020.

[34] Kevin Lu et al. *Pretrained Transformers as Universal Computation Engines.* 2021. arXiv: 2103.05247 [cs.LG].

[35] Dezhao Luo et al. *Video Cloze Procedure for Self-Supervised Spatio-Temporal Learning.* 2020. arXiv: 2001.00294 [cs.CV].

[36] Shuang Ma et al. *Contrastive Learning of Global and Local Audio-Visual Representations.* 2021. arXiv: 2104.05418 [cs.LG].

[37] Antoine Miech et al. *End-to-End Learning of Visual Representations from Uncurated Instructional Videos.* 2020. arXiv: 1912.06430 [cs.CV].

[38] Antoine Miech et al. *HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips.* 2019. arXiv: 1906.03327 [cs.CV].

[39] Pedro Morgado, Nuno Vasconcelos, and Ishan Misra. *Audio-Visual Instance Discrimination with Cross-Modal Agreement.* 2021. arXiv: 2004.12943 [cs.CV].

[40] Mehdi Noroozi and Paolo Favaro. "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles". In: *CoRR* abs/1603.09246 (2016). arXiv: 1603.09246. URL: http://arxiv.org/abs/1603.09246.

[41] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. *Representation Learning by Learning to Count.* 2017. arXiv: 1708.06734 [cs.CV].

[42] Tian Pan et al. *VideoMoCo: Contrastive Video Representation Learning with Temporally Adversarial Examples.* 2021. arXiv: 2103.05905 [cs.CV].

[43] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32.* Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[44] Mandela Patrick et al. *Multi-modal Self-Supervision from Generalized Data Transformations.* 2020. arXiv: 2003.04298 [cs.CV].

[45] Mandela Patrick et al. *Support-set bottlenecks for video-text representation learning.* 2021. arXiv: 2010.02824 [cs.CV].

[46] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *the Journal of machine Learning research* 12 (2011), pp. 2825–2830.

[47] Rui Qian et al. *Spatiotemporal Contrastive Video Representation Learning.* 2021. arXiv: 2008.03800 [cs.CV].

[48] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision.* 2021. arXiv: 2103.00020 [cs.CV].

[49] Prajit Ramachandran, Barret Zoph, and Quoc V Le. "Searching for activation functions". In: *arXiv preprint arXiv:1710.05941* (2017).

[50] Andrew Rouditchenko et al. *Self-Supervised Audio-Visual Co-Segmentation.* 2019. arXiv: 1904.09013 [cs.CV].

[51] Sandvine. *The Global Internet Phenomena Report.* 2020.

[52] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. "UCF101: A dataset of 101 human actions classes from videos in the wild". In: *arXiv preprint arXiv:1212.0402* (2012).

[53] Chen Sun et al. *VideoBERT: A Joint Model for Video and Language Representation Learning.* 2019. arXiv: 1904.01766 [cs.CV].

[54] Li Tao, Xueting Wang, and Toshihiko Yamasaki. *Pretext-Contrastive Learning: Toward Good Practices in Self-supervised Video Representation Leaning.* 2021. arXiv: 2010.15464 [cs.CV].

[55] Yonglong Tian et al. *What Makes for Good Views for Contrastive Learning?* 2020. arXiv: 2005.10243 [cs.CV].

[56] Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. *Unsupervised Learning of Video Representations via Dense Trajectory Clustering.* 2020. arXiv: 2006.15731 [cs.CV].

[57] RB Tootell, MS Silverman, and RL De Valois. *Spatial frequency columns in primary visual cortex.* 1981 Nov 13. DOI: 10.1126/science.7292014.

[58] Michael Tschannen et al. *On Mutual Information Maximization for Representation Learning.* 2020. arXiv: 1907.13625 [cs.LG].

[59] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. "Self-supervised Video Representation Learning by Pace Prediction". In: *CoRR* abs/2008.05861 (2020). arXiv: 2008.05861. URL: https://arxiv.org/abs/2008.05861.

[60] Jiangliu Wang et al. *Self-supervised Video Representation Learning by Uncovering Spatio-temporal Statistics.* 2021. arXiv: 2008.13426 [cs.CV].

[61] Jinpeng Wang et al. *Self-supervised Temporal Discriminative Learning for Video Representation Learning.* 2020. arXiv: 2008.02129 [cs.CV].

[62] Lei Wang, Piotr Koniusz, and Du Q. Huynh. *Hallucinating IDT Descriptors and I3D Optical Flow Features for Action Recognition with CNNs*. 2019. arXiv: 1906.05910 [cs.CV].

[63] Tongzhou Wang and Phillip Isola. *Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere*. 2020. arXiv: 2005.10242 [cs.LG].

[64] Chen Wei et al. "Iterative Reorganization with Weak Spatial Constraints: Solving Arbitrary Jigsaw Puzzles for Unsupervised Representation Learning". In: *CoRR* abs/1812.00329 (2018). arXiv: 1812.00329. URL: http://arxiv.org/abs/1812.00329.

[65] Donglai Wei et al. "Learning and Using the Arrow of Time". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8052–8060.

[66] Mike Wu et al. *On Mutual Information in Contrastive Learning for Visual Representations*. 2020. arXiv: 2005.13149 [cs.LG].

[67] Dejing Xu et al. "Self-Supervised Spatiotemporal Learning via Video Clip Order Prediction". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[68] Ceyuan Yang et al. *Video Representation Learning with Visual Tempo Consistency*. 2020. arXiv: 2006.15489 [cs.CV].

[69] Yuan Yao et al. *Video Playback Rate Perception for Self-supervisedSpatio-Temporal Representation Learning*. 2020. arXiv: 2006.11476 [cs.CV].

[70] Chengxu Zhuang et al. *Unsupervised Learning from Video with Deep Neural Embeddings*. 2020. arXiv: 1905.11954 [cs.CV].